

## Proposed Luminosity Accounting/Tracking Procedure

23-April-1999, RDS

There will be a **luminosity monitoring process** running on the online system. It may have connections to other DAQ processes which depend on run status, but it will monitor luminosity **independent of data runs** in order to supply a continuous stream of data to the accelerator. To do this **it will query TCC** (trigger control computer) at a rate not to exceed 1 Hz. The actual rate will be selected to have sufficient counting rate not to be dominated by statistical fluctuations, while maximizing feedback to the accelerator. In order to allow TCC to capture the required information, independent of run status, the framework will be "paused" for about 4  $\mu$ Sec for each request. The meaning of **pause** is that there will be no level 1 accepts issued during this time. Live time for active trigger sets will be properly accounted for. The scalers used for this monitoring are never reset, and the raw numbers are sent to the luminosity monitoring process. It is the responsibility of the luminosity monitoring process to remember previous scaler values to take the appropriate differences. In the block of data sent to the luminosity monitoring process there will also be the following numbers:

- 1) A 32 bit "*luminosity index*" number,
- 2) An 8 bit "*tick*" number,
- 3) A 32 bit "*turn*" number, and
- 4) A 64 bit "*date-time stamp*".

The definition of the **luminosity index** number is a number which is set to one(1) until Jan 1,2000 00:00:01. After that it is incremented by one whenever the luminosity monitoring process requests from TCC a luminosity update WITH index update. The data shipped on that update is the final set of scaler readings for the old index number, and all data (including raw data) after the turn/crossing listed will be marked with the next index. Whenever the index number is changed an entry into a luminosity database containing all relevant information for the just ended luminosity interval must be made. This index number is maintained in the trigger framework, with a copy in non-volatile memory so it is remembered even through power outages. It is expected that this index will only be updated once every 5 minutes or so, but there is no restriction on when it is updated.

The definition of the *tick* number is a number which ranges from 0-158 and increments every 7 rf buckets (132 nSec). It is synchronized to the accelerator and indicates which of the potential 159 locations for a proton bunch is currently in front of the D0 detector.

The definition of the *turn* number is a number which increments one for every time the same beam particle could go by D0. This is NOT the turn number used in stamping different parts of the same event, which gets reset on SCL init commands. This number only resets on power failures.

The definition of the *date-time stamp* is a packed BCD data-time value identical to the format used by all geographic sectors to time stamp locally detected significant events (including errors). For details on the format see the D0 DAQ Geographic Section Specifications on the web. It is not meant to be more accurate than a few minutes (although typically should be good to a second or two). To automatically insure at least this level of accuracy, I propose that a standard TCP time server be setup (either in D0 or elsewhere on site) where all network accessible systems can check their local time, a few times per day.

## Numbers that could be used to identify events

The following numbers are (or could be) stamped on every event (by the trigger framework):

- 1) Luminosity Index (32 bits)  
Should be used to split high rate streams into multiple files for a single run. Each file is then associated with a beginning and ending index number. Different streams can split at different index values. This works assuming that the index number is incremented often enough.
- 2) Coor Run Number (32 bits)  
These 32 bits are fully under the control of Coor and can be updated whenever data taking is paused. A suggested use of these bits is a global run number. Remember that because multiple run, containing different trigger bits are allowed to run simultaneously, but this number can not be different as a function of trigger bits. If this number changes on every begin/end run, then Coor can store all global run numbers used during a particular local run.
- 3) 132ns Clock (32 or 64 bits)  
This number increments about 7 million times per second and is only reset on power failures. This insures that the number for any run starts at its smallest value and continually increases until the end of the run. It will be the same if a single event occurs in two separate (but simultaneous) runs. In a typical 4 hour run this number changes by about  $1.1 \times 10^{11}$ , which is a large number but not much worse than 131.225.224.121. The full 64 bit number is large enough that I will not live to see it overflow, and therefore is unique in my lifetime.
- 4) L1 accept number (32 bits)  
This number is similar to but different from the L1 accept number used by the geographic sectors, in that it only resets on power failure (and possibly a request from Coor). At a 10KHz rate this number overflows about once every 100+ hours and is therefore unique and monotonic in any run.
- 5) L2 accept number (32 bits)  
This number is identical to the L3 transfer number, except 32 bits long instead of the 16 lsb's used by the geographic sectors. It is reset on power failure (and possibly a request from Coor). At a 1KHz rate this number overflows about once every 1000+hours and is therefore unique and monotonic in any run.
- 6) Date-Time number (12 byte BCD)  
This is the standard geographic sector time stamp. It is only precise to a second and accurate to a few seconds, so can not be used by itself to uniquely identify an event.

## Other noteworthy observations

Any number assigned after the events enter the L3 farm can no longer be in chronological order. In fact you can not even assign a number which is not mixed across stream parts, because we will be splitting files on a luminosity index number boundary.

The present plan is to NOT write an event more than once per run. This requires analysis programs to access multiple streams to get all the data for a given stream, but eliminates the need to check if the event exists in a different file. To hold down the number of different streams, it is expected that a few of the lower rate streams will be combined into one "stream". This implies that analysis programs running on these "mixed streams" will have to check to see if the event they are reading actually passes the particular trigger bits they are interested in.

I have not been able to understand where the stream assignment is made, as different people appear to think it is made at different levels. My personal opinion is that it should be made in the L3 output path, as the event is shipped off over the net. The Level 3 should also define a small header block containing all the information needed to properly handle the data. During the online meeting it was made clear that the L3 framework will handle the stream assignment, before the data leaves to node. If multiple runs are in progress, this might entail defining more than one stream destination for an event.

During the online meeting it was noted that there should be a restriction placed on multiple runs which are taking data simultaneously. Since it is expected that events that have trigger bits from different runs will be written to multiple streams, the stream name should not be allowed to be identical for simultaneous runs. Coor should enforce this restriction so that other subsystems can assume this will never happen. Since this means that the same event can appear in different files, but only if they are associated with different runs, no analysis should attempt to combine data taking in different runs which are taken during the same luminosity index number, since one can not correctly calculate the luminosity for that mixed exposure.

Also during the online meeting it was noted that if the full bandwidth of the level 3 system was directed to a single stream, it would fill the design file size (1Gb) in a minute or so. Since we were expecting to only close files on luminosity index number changes, this would require an index number update at about one per minute (instead of the proposed once per 5 minutes. If logging the luminosity data once every minute is not practical, more than just the luminosity index number must be checked and logged for deciding when one is allowed to close a stream file. Also restricting files to only close on luminosity index number changes, means that the luminosity monitoring process must be running whenever data is being logged, even if the accelerator is not running. Since we can still recover the integrated luminosity that occurred while the luminosity monitoring process was down, we should not require the data to stop if the luminosity monitoring process goes away. On the other hand, for runs which need luminosity recorded, runs should not be able to start until the luminosity monitoring process indicates that it has successfully incremented the luminosity index number. Also end runs which are marked with needing luminosity should insure that the luminosity index is changed at the instant that trigger accepts are disabled for trigger bits in that run.